Postel

Title: Host-Imp Interface

Author: G. Deloche

Installation: University of California at Los Angeles

Date: May 1969

Network Working Group Request for Comment: 7

G. Deloche → Prof J. Estrin
Prof L. Kleinrock
Prof B. Bussel
D. Mandell
S. Cracker
L. Bonamy

Object : Arpa Network - Specification outlines :
HOST-IMP (HI) interface programs.

I Introduction

II Scope of the software organization
  II-1 Network program
  II-2 Handler program

III Questions

I Introduction

This paper is concerned with the pr[illegible] software design of the HOST IMP interface. Its main purpose is on the one hand to [illegible] functions that will be implemented, and on the other hand to provide a base for [illegible] on [illegible]

This study is carried out [illegible] a [illegible] contract n° 162.

II Scope of the software organization.

The system is based upon two main programs: the Handler program that drives the channel hardware unit, and the Network program which carries out the user's transmission requests.

As the communication is full duplex, each of these programs can be viewed as divided into two parts: one is concerned with the output data, the other with the input. (See fig. 1)

These two programs echange data through a pool of buffers, and logical information through an interface table.

In the following we only focus on the output part of each program (See fig 2). The input part would be very similar.

II-1 Network program.

II-1-1. Multiplex function

This program multiplexes the outgoing messages (and distributes the incoming messages). The multiplexing consist in stacking up all the user's (or caller, or party) requ and filling up the pool of buffers so as to keep the handler busy emitting.

Multiplexing (and distrubution) is based on the lin identification numbers. (link = logical connection betw two users). The multiplexing problem is closely related to the interface between an user's [illegible] and the the network program, that is in fact operating system (See below Questions)

II-1-2 Output messages processing.

When an user's program wants to send out text it should indicate the following informat (through a macro, or as call parameters): text loca text lenght in byts, and destination.

Using these data the Network program:

* prepares a 16 bits Host heading (1 bit: trace, [illegible] bits: spares, 8 bits: link identification n°, 5 bits: dest host)

* inserts a 16 bits marking between the hea and the text so as to start the text at a word boundary. This marking consists of a one preceding the first bit of the text and, in turn, preceded by fifteen zeros to fill up the gap.

* checks the lenght of the user's text. If it exc 1006 byts

$$\left[\frac{8080\,(\text{max Host message lenght}) - 32\,(\text{heading + marking})}{8\,(\text{byte} = 8 \text{ bits})}\right.$$

the program breaks down the text into a sequence of mess whose maximum lenght is 1006 byts. Each of these me

is preceded by a heading as explained above.

Remark : in that case one of the heading spare bits ca be used for indicating that several messages belong to the same tests.

* *transcodes* the EBCDIC characters constituting the messages into ASCII characters.

* *fills* the buffers of the pool with the contents of the messages.

* *updates* the content of the interface table and moves the filling pointers (see below)

II - 2 Handler program.

This program is initiated either by the network program, or by the I/O interrupt.

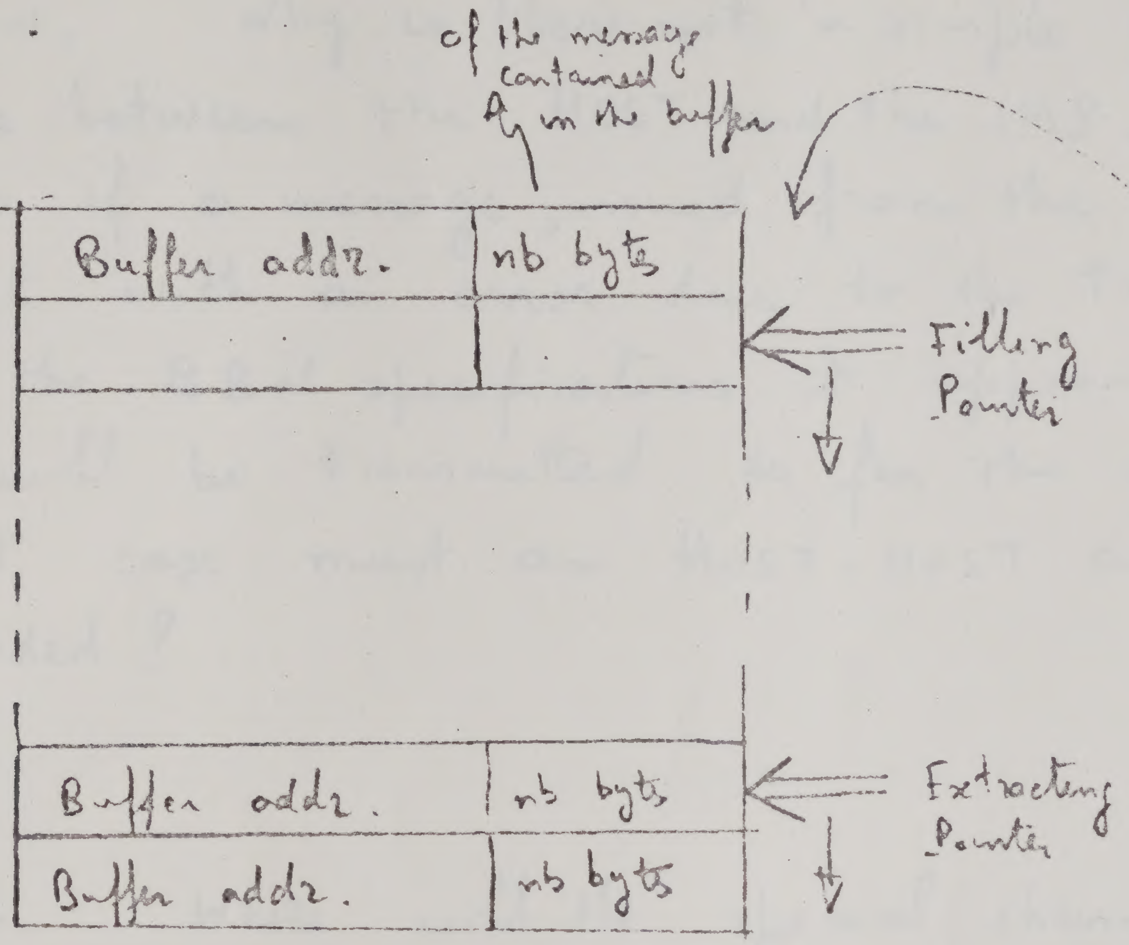
This program will be very short. It will be coded in master mode (privileged instructions) and should be integrated in the I/O supervisor of the operating sys

This program :

* *controls* the channel hardware unit. It initia the emission, eventually provides data chaining between the buffer tests the different device status upon receiving an interru

* *empties* the buffers that are filled up by

This table could be a ring table with 2 pointers. one for filling, the other for extracting. they are respectively updated by the network and the handler program.

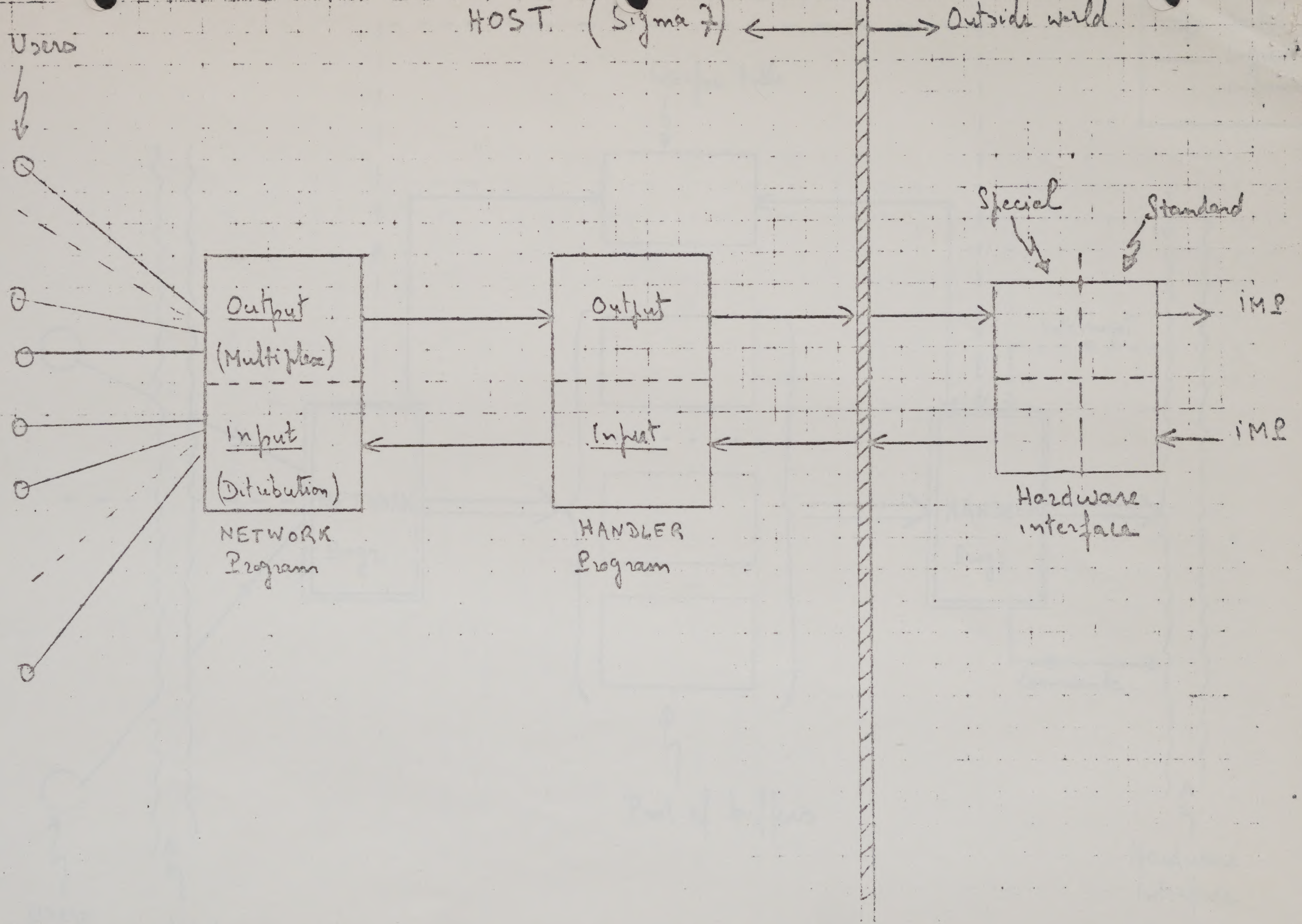

of the message
contained
lg in the buffer
Buffer addr.
nb bytes
Filling
Pointer
Buffer addr.
nb bytes
Extracting
Pointer
Buffer addr.
nb bytes

III Questions

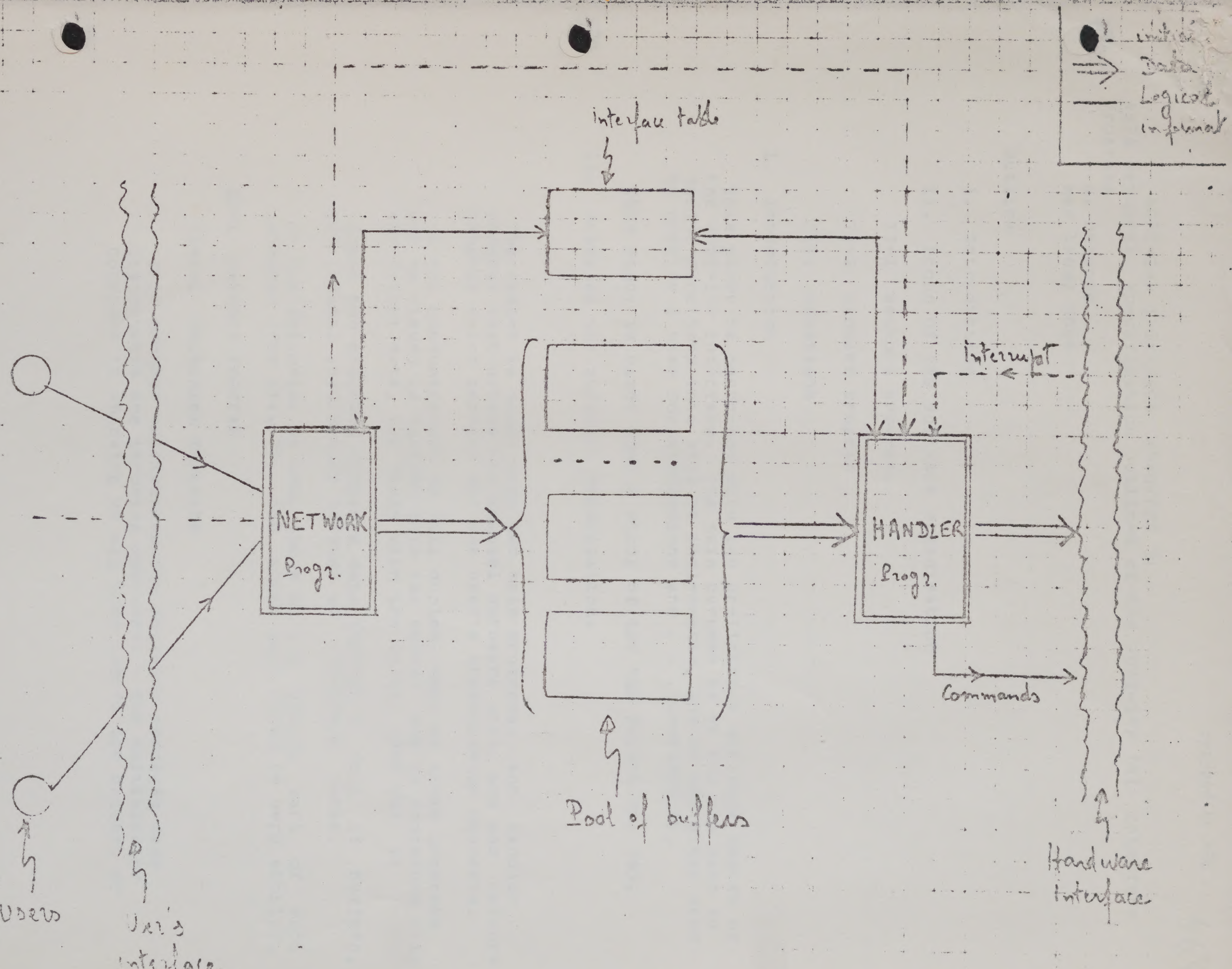
III-1. Why is there not a simple control procedure between the HOST and the IMP? What happens if a message, issued from the HOST, reaches the IMP with an error due to the transmission?
From the BBN specifications it appears that this error will be transmitted as far the receiving HOST.
In that case must an HOST-HOST control procedure be provided?

III-2. Where will the special channel hardware unit be connected (MIOP/SIOP)?
How will this device be notified of an outgoing message end in order to start the padding?
(The program will provide to the MIOP SIOP the number of bytes of the outgoing message, and will receive back an interrupt when the last byte is sent out.
Is it that signal which will be also sent to the special device?)
Vice versa how does the Handler know the lenght of the incoming message? From the content of the previous one or should this program always ready to receive a mess of maximum lenght? (then an interrupt should be trigge when the real end is detected by the hardware)

III-3 When does the Cordo documentation will be available in order to design the user-network program interface. What are the mecanisms for program initiations, transfering parameters from one program to another etc..


HOST (Sigma 7)
Outside world
Users
Output
(Multiplex)
Input
(Distribution)
NETWORK
Program
Output
Input
HANDLER
Program
Special
Standard
Hardware
interface
IMP
IMP

(Fig 1)


initiat.
Data
Logical informat
interface table
NETWORK Progr.
HANDLER Progr.
Interrupt
Commands
Pool of buffers
Users
User's interface
Hardware interface

(Fig. 2)

retyped by ARC

4693

HOST-IMP INTERFACE (NWG/RFC 7)

ARPA Network: Specification Outlines of the HOST-IMP (HI) Interface Programs

G. Deloche
May 1969; UCLA

Outline

- I. Introduction
- II. Scope of the software organization
  - II-1 Network program.
  - II-2 Handler program
- III. Questions

I. Introduction

This paper is concerned with the preliminary software design of the HOST-IMP interface. Its main purpose is on the one hand to define functions that will be implemented, and on the other hand to provide a base for discussions and . . .(unreadable).

This study is based upon a study of the BBN Report No. 763.

II. Scope of the software organization.

The system is based upon two main programs: the Handler program that drives the channel hardware unit, and the Network program which carries out the user's transmission requests.

As the communication is full duplex, each of these programs can be viewed as divided into two parts: one is concerned with the output data, the other with the input. (See fig. 1)

These two programs exchange data through a pool of buffers, and logical information through an interface table.

In the following we only focus on the output part of each program (see fig. 2). The input part would be very similar.

II-1 Network Program

II-1-1 Multiplex function

This program multiplexes the outgoing messages (and distributes the incoming messages) - The multiplexing consists in stacking up all ther user's (or caller, or

party) . . .(unreadable) and filling up the pool of buffers so as to keep the handler busy emitting

Multiplexing (and distribution) is based on the link identification numbers. (Link - logical connection between two users). The multiplexing problem is closely related to the interface between a user's program and the network program, that is in fact . . .(unreadable) operating system (see below: Questions)

II-1-2 Output message processing

When a user's program wants to send out text it should indicate the following information (through a macro, or as call p parameters): text location, text length in bytes, and destination-

Using these data the Network program:

* prepares a 16 bit Host Leading (1 bit: trace, 2 bits: spares, 6 bits: link identification no., 5 bits: destination host)

* inserts a 16 bit marking between the header and the text so as to start the text at a word boundary. This marking consists of a one preceding the first bit of the text and, in turn, preceded by fifteen zeros to fill up the gap.

* checks he length of the user's text - if it exceeds 1006 bytes [.xxx.] the program breaks down the text into a sequence of messages whose maximum length is 1006 bytes - Each of these messages is preceded by a heading as explained above.

> Remark: in that case one of the heading spares could be used for indicating that several messages belong to the same text.

* transcodes the EBCDK characters constituting the messages into ASCII characters.

* fills the buffers of the pool with the contents of the messages.

* updates the content of the interface table and moves the filling pointer (see below)

II-2 Handler Program

This program is initiated either by the network program, or by the I/O interrupt

This program will be very short. It will be coded in master mode (privileged instructions) and should be integrated in the I/O supervisor of the operating system.

This program:

- * controls the channel hardware unit. It initiates the emission   * controls the channel hardware unit. It initiates the emission, eventually provides data chaining between the buffers, tests the different device status upon receiving an interrupt

  empties the buffers that are filled up by the network program.

- * explores and updates the interface table (see below)

- * can eventually insure a control transmission procedure with the IMP (See Questions)

II-3 Buffers and Interface Table.

II-3;1 Buffers.

They should be large enough for containing the maximum host message text + heading and marking (1006 + 4 = 1010 bytes)

Consequently the buffer size could be chosen equal to 256 words (1024) bytes). As for the buffer number it will determine by the link utilization frequency -

II-3-2 Interface Table

It is through this table that the network program informs the handler with the location and length of the emitting data.

This table could be a ring table with two pointers: one for filling, the other for extracting. They are respectively updated by the network and the handler programs.

*****

III. Questions

III-1 Why is there not a simple control procedure between the host and the IMP? What happens if a message, issued from the HOST, reaches the IMP with an error due to the transmission?

From the BBN specifications it appears that this error will be transmitted as far as the receiving HOST.

In that case must an HOST-HOST control procedure be provided?

III-2 Where will the special channel hardware unit be connected (MIOP/SIOP)?

How will this device be notified of an outgoing message end in order to start the padding?

(The program will provide to the MIOP/SIOP the number of bytes of the outgoing message, and will receive back an interrupt when the last byte is sent out. Is it that signal which will be also sent to the special device?)

Vice versa how does the Handler know the length of the incoming message? From the contents of the previous one or should this program always be ready to receive a message of maximum length? (then an interrupt should be triggered when the real end is detected by the hardware).

III-3 When does the Gordo documentation will be available in order to design the user-network program interface. What are the mechanisms for program initiations, transferring parameters from one program to another etc.

Fig. 1

Fig. 2